

Requirements for an Enterprise Application Integration Tool

Kimmo Röppänen

University of Tampere
School of Information Sciences
Computer Science
M.Sc. thesis
Supervisor: Timo Poranen
May 2013

University of Tampere

School of Information Sciences

Computer Science

Kimmo Röppänen: Requirements for an Enterprise Application Integration Tool

M.Sc. thesis, 50 pages

May 2013

As companies try to keep up with the competition they often rely on information technology (IT) to make the business work more efficiently and cheaper. As the amount of IT and software increases the companies need to utilize different applications to communicate with each other. Otherwise same data would need to be inserted into each application separately, thus making the business work less efficiently. To help with the communication problem many different kinds of applications have been created. These tools are called Enterprise Application Integration tools. The problem is that the number of tools is very large and finding the most suitable tool can be an overwhelming task.

This study presents a set of qualities in software that are important in Enterprise Application Integration tools. As a research method this study adopts a case study of a company trying to find a tool to help integration process with an ERP product. Some recommendations for a suitable tool are given as a conclusion of the case study.

The study ends in general conclusions by expanding the conclusions made in the case study to support a more general decision making process. Main conclusion is that there are 9 qualities of an EAI product that should be paid attention to. These qualities are: Stability, Changeability, Security, Performance/Scalability, Connectivity, Management/Monitoring, Message Transformation, Routing and Pricing.

Key words and terms: Enterprise Application Integration, EAI, Asynchronous, Messaging, ESB, Enterprise Service Bus, Integration.

Contents

1.	Introduction	1
1.1.	Improving efficiency	1
1.2.	Research questions	1
2.	Enterprise Application Integration.....	3
2.1.	File Transfer	5
2.2.	Shared Database	5
2.3.	Remote Procedure Invocation	6
2.4.	Messaging.....	7
2.5.	Web Services	8
2.6.	Point-to-Point	9
2.7.	Broker.....	10
2.8.	Enterprise Service Bus	12
3.	Enterprise Application Integration tools.....	14
3.1.	Software Quality	15
3.2.	Commercial EAI Tools	17
3.2.1.	IBM WebSphere Message Broker	17
3.2.2.	Mule ESB Enterprise	19
3.3.	Free Open Source EAI Tools	22
3.3.1.	WSO2 ESB	22
3.3.2.	Apache ServiceMix	23
3.3.3.	Mule ESB Community	24
3.4.	Enterprise Resource Planning	25
4.	Case Study: Glass & Frame.....	27
4.1.	Merit Consulting Finland.....	27
4.2.	Jeeves	27
4.3.	The Customer.....	28
4.4.	G&F integration needs.....	29
4.5.	Requirements	31
4.5.1.	Stability.....	32
4.5.3.	Security	33
4.5.4.	Connectivity	35
4.5.5.	Performance/Scalability.....	35
4.5.6.	Management/Monitoring.....	36
4.5.7.	Message Transformation	36
4.5.8.	Routing	37
4.5.9.	Pricing.....	38
4.6.	Tool Comparison.....	38

4.6.1.	IBM WebShpere MQ.....	38
4.6.2.	Mule ESB (Enterprise and Community)	39
4.6.3.	WSO2	40
4.6.4.	ServiceMix	40
4.7.	Recommendation for suitable tool	42
5.	Conclusions	43
	References	45

1. Introduction

In enterprise information technology (IT) companies commonly have multiple applications that need to share data between each other. Whether it is warehouse management system telling order processing if the ordered items are in stock or accounting needing to know from invoicing which bills are sent and which are not. Many times this sharing of information is done in a very inefficient way, for example people sending Excel sheets to each other. So how could we make the process of data sharing more efficient?

1.1. Improving efficiency

When companies add new applications into their IT architecture they need to make them communicate with the other existing applications so as the data, and sometimes functionality, can be shared between them. There are multiple ways of doing this and often what is good way for one application might not be possible for another. Since applications have probably been acquired during the complete lifespan of the company, the tools provided for the communication might be very mixed in style and technology. This is where the study of Enterprise Application Integration (EAI) and tools that have been specifically developed with EAI in mind come into the picture. The purpose of Enterprise Application Integration is to make the process of integration simpler and easier by providing the tools for connecting to applications, transforming and routing messages and monitoring the message traffic.

1.2. Research questions

Choosing the right EAI tool for a company can be difficult. If it's a small company with needs for very minimum of integrations to other systems then should it just rely on very simple point to point integration or should they prepare to a growing need for integration and acquire some kind of EAI tool? If so, then what is the right tool for their needs? What requirements the tool needs to fulfil? What kind of future integrations they should be prepared for? How much money can the company afford to spend on a single EAI tool? What about Open Source tools? Would those offer everything that is required and how reliable they are? How much more work is needed to configure and maintain the tool? This paper does not answer all of these questions but concentrates on finding the general requirements areas that should be considered when an EAI tool selection process is initiated. Also this paper recommends some suitable EAI tools in the specific case of Merit Consulting Finland.

In this thesis we look into some of the common Enterprise Application Integration tools and compare them to each other. The comparison is done by creating a case study using a company called Merit Consulting and trying to find the most suitable tool for their use. The findings are then expanded to provide suggestions on what are the things that should be held important when trying to find the correct tool for any company. The method of research was chosen as the author himself is working in Merit Consulting and much of the data needed was already available. Also this method was seen as the most likely to give the best results from which Merit could make their decision. The research could have also been conducted by reviewing the literature about the integration domain but this might not have provided enough information for the specific case of Merit Consulting. On the other hand a more extensive review on literature about integration might have provided a more complete generalization of the requirements for EAI tool selection.

Merit Consulting Finland [Merit FI, 2013] is in search for a new EAI tool to ease their job of integration with a new Enterprise Resource Planning (ERP) system they have acquired into their portfolio. This new ERP system does not have very extensive integration tools and so Merit is trying to find an external tool to simplify the integration process. Also the tool might help their customers in other integration tasks in the future.

Enterprise Application Integration is discussed more in Chapter 2. Some general software quality issues and the different commercial and open source tools are presented with some depth in Chapter 3. Also a brief introduction to enterprise resource planning systems is provided in this chapter. In Chapter 4 we look into the case study of Merit Consulting and decide what would be a good tool for them. Chapter 5 goes through more general conclusions about how companies should decide what the best solution is for them. Also some limitations of validity to the results are explained in this chapter.

2. Enterprise Application Integration

The term Enterprise Application Integration (EAI) first appeared in the USA around 1996-1997 with the publications of articles or research notes by large analyst firms [Manouvrier & Menard, 2010]. According to Manouvrier and Menard [2010] the definition of Enterprise Application Integration is *a collection of methods, tools, and services that work together to bring heterogeneous applications into communication, as part of the traditional, distributed or extended enterprise*. It is important to notice that not only is data shared by applications but they can also share some of their functionalities with other applications.

In the past at a time before EAI was introduced companies had many different applications doing various functions that were required by their business plan. The problem was how to make these applications share their data since the same information could be needed by many applications. At first all the data was shared by people copying the data from application to application. After a while this was seen as too slow and an error prone mechanism for data sharing so companies started connecting different applications to each other. This is called Point-to-Point (PtP) - pattern (Subsection 2.1.6). In Figure 1 is illustrated how the point-to-point architecture looked like.

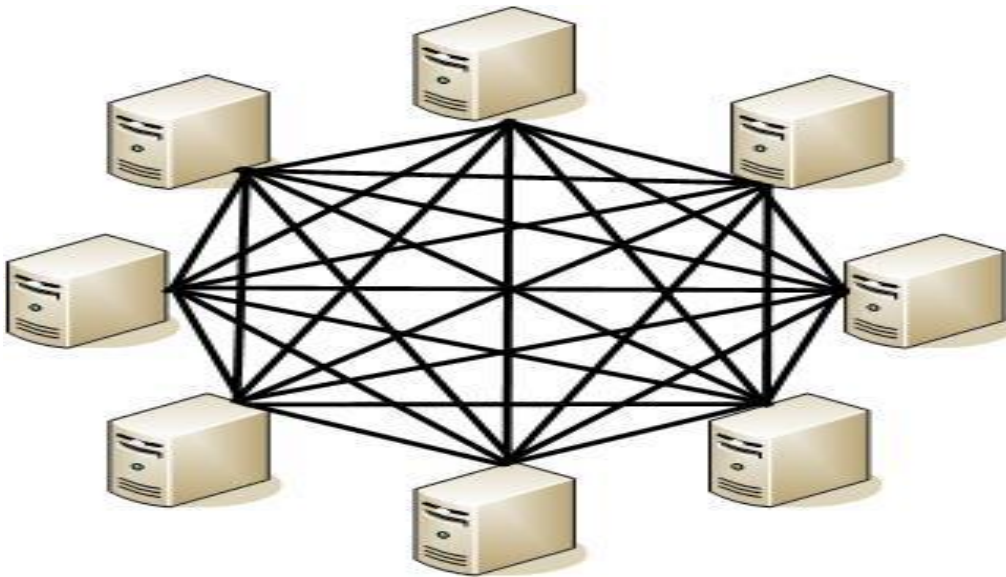


Figure 1: Point-to-Point architecture [Brooks-Bilson, 2007].

As the number of applications increased, the points of integration between different applications also increased. Due to the n -rule (Subsection 2.1.6) each new application added to the infrastructure also introduced multiple new integration points. The more points there were the more difficult it was to manage the enterprise's IT architecture.

After a while the IT infrastructure with its numerous PtP-integrations became too complex to maintain and another way of integration was needed. This is where EAI applications come into play. EAI applications are tools that enable integration of multiple applications without tightly coupling them together. In Figure 2 is illustrated the place for an EAI tool in the enterprise IT infrastructure. EAI incorporates functionality from disparate applications and leads to cheaper, more functional and manageable IT infrastructures [Themistocleous et al., 2004].

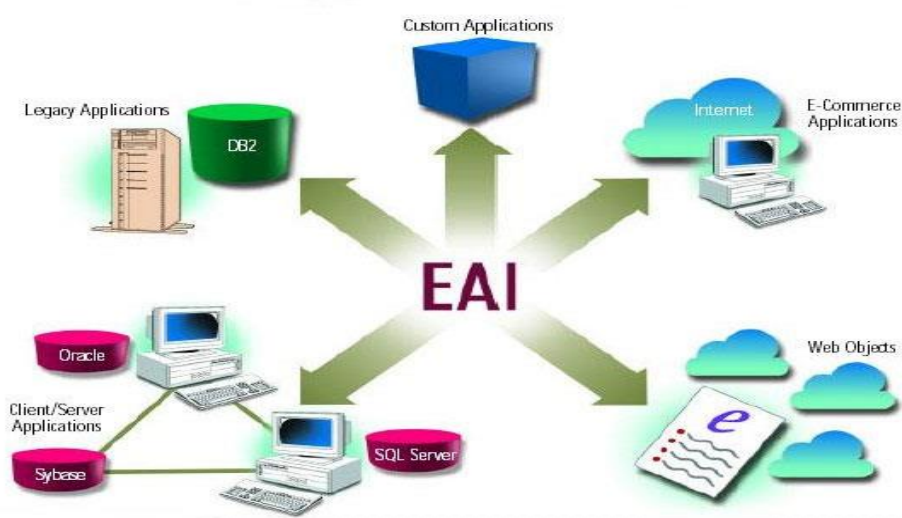


Figure 2: Enterprise Application Integration [Cymatric, 2011].

Integration patterns are tried and tested ways of solving problems that many application developers have faced during software projects involving integrating two applications together. They are not cut and paste-type ready-made solutions to a specific problem but more of a guideline on how some problematic situations can be solved. It's up to the developer to actually choose the correct patterns and implement a specific solution for his/her specific situation. When creating an EAI tool it is good to have a wide general knowledge of different patterns and how they can be used so that it can be taken into account when designing the tool. Of course trying to implement every possible pattern solution to the EAI tool is close to impossible and also impractical. Some integration patterns will always be left out of the EAI tool since they might be impossible to implement with other chosen patterns or might just never be needed. But it is also good to acknowledge what the application is not capable of as it is to acknowledge what it is capable of. Umapathy and Burao [2005] have stated that *Enterprise Integration Patterns provide designers integration tactics and strategies in the form of design patterns*.

2.1. File Transfer

In file transfer the communication between applications is handled by transforming data in one application into a previously agreed file format, saved and sent to a location from where the other application can pick it up and process the data. Great advantage of file transfer is that it, in most cases, is platform and language independent way of communication so applications don't need to have any extra tools or packages to enable the integration [Hohpe & Woolf, 2005]. Also integration with files is asynchronous so the receiving application doesn't need to know who sent the file, just how the file should be processed. Decoupling applications makes the job of integration much easier since any changes in one application does not affect the other applications as long as the file format stays the same. Another good thing with file transfer is that, as there's actually no direct interface that the applications provide but everything goes through files then it's much harder for malicious software to cause major damage to the applications.

As any style of integration, file transfer is not without problems. First, the resource cost of creating/processing a file and transferring it through the network is relatively high. If the applications need to communicate in an extremely high frequency then file transfer might become a bottleneck in the integration. Another issue with this style of communication is that the updates in an application tend to happen infrequently but the file transfer is scheduled to happen only at certain times. So when and how often the files should get sent so that data stays synchronized in both applications? As said earlier the more frequently you transfer the data the more expensive it gets on the computer and network resources [Hohpe & Woolf, 2005]. Luckily this is not as big of an issue with modern powerful computers and fast networks as it was some years ago.

2.2. Shared Database

A problem with file sharing is that there will always be moments when the data is not synchronized between applications. For modern businesses this might become a critical problem. What if customer changes his/her address in one application and then makes an order to a salesperson that sets the delivery address from another application? If the information of the change has not yet been updated in this application then the order will end up in wrong address. By sharing a database all integrated applications can be certain that the data is always consistent [Hohpe & Woolf, 2005]. However, it's close to impossible to find two applications that can fully use the same database so what sharing database usually means is that there is an external data storage that both applications are connected to for the purpose of sharing their data.

As said earlier, one advantage of shared database is that it provides consistent data to all applications by enabling the possibility of real-time updates to the database. Also with this integration style it can be certain that the data is used in the same format everywhere. Many integration problems come from the fact that different systems look at the data in different ways [Hohpe & Woolf, 2005]. Since SQL [SQL, 2013] and relational databases are widely used in software it is quite certain that any application can connect to the database without the need to include a new technology to the mix.

Issues with shared database come up when designing a suitable schema that all applications can conform to. This can easily lead to a too complex and difficult schema to work with and can cause some critical applications to separate from the shared database [Hohpe & Woolf, 2005]. Many commercial software packages might also not work at all or might work with limited functionality with any other database schemas than their own making the sharing impossible. Even if unified schema could be found for all applications and they could all use it perfectly, there is the issue of how to manage multiple applications reading and modifying the database frequently? With many applications connecting to database trying to read, insert, and update the data, it can quickly cause deadlocks preventing others from accessing the data. Also if applications are distributed across a wide area network, accessing the database can get too slow. The database could of course be copied to multiple servers so applications could access the database that is closest to them but then there would be need to figure out how to handle updates between these databases.

2.3. Remote Procedure Invocation

File transfer and shared database are suitable styles of application integration when only the data of the applications need to be shared. But what if one application would like to finish its processing, but to do this it would need to wait for someone to do some data handling in another application? The application could connect to the other application's database and try to change the data the way it would have been done by the other application. Unfortunately very often when application processes its data, it has a ripple effect around the database affecting multiple different columns. If another application tampers with the database there is no guarantee that the all the affected database columns will be changed accordingly. File transfer would require the receiving application to handle the data. Since file transfer is asynchronous, the processing would be delayed. Also how would the waiting application know when it can proceed? With Remote Procedure Invocation (RMI) (or Remote Procedure Call (RPC)) applications can call each other's exposed functionality and take advantage of direct communication with the application. With RMI the applications are only responsible of the data in their

own database and communication is synchronized making the responses almost immediate. Figure 3 explains how RMI works.

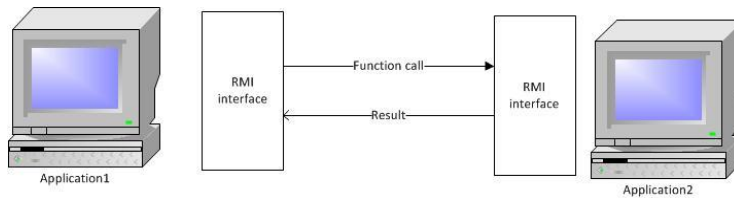


Figure 3: Remote Procedure Invocation.

However, RMI makes applications become tightly coupled to each other and a major failure in one application would affect the other application as well making them both fail. Because RMI is very close to the normal procedure calls, it is very simple to understand by most developers but it also means that developers might have difficult time understand the differences in performance and reliability between local procedure calls and Remote Procedure Calls. Failure to understand this difference will easily lead to slow and unreliable system. [Hohpe & Woolf, 2005]

Although RMI is mainly thought of as synchronic communication, asynchronous RMI communication is also possible. The calling application could call the remote function from another thread and continue to other work in its main thread. Although this would still require the applications to be tightly coupled since the second thread would still need to stay connected to the remote application and wait for its processing to finish. [Rinneheimo, 2008]

2.4. Messaging

File transfer and shared database are good options when you only need to share the data between applications. But file transfer is somewhat heavy on resources and keeping data synchronized is difficult. On the other hand with shared database it is difficult to find a suitable schema for all applications to be able to use it. Also many applications do not work with any other than their own database (DB) schemas. When you need to share functionality Remote Procedure Invocation is a viable solution. With RMI you have the applications tightly coupled to each other making application execution and application development more unreliable. What if you needed to have fast communication between applications sharing data but couldn't couple them together or to any common database schema?

Messaging is an asynchronous way for applications to share data between each other without the need for sender to even be aware of the receiving application. It only needs to send a message to a message channel and the channel will make sure the receiving

application will receive it. Even if the receiver is not online when the message is sent the message channel can queue the messages and wait for the receiver to get online. The messaging allows much more decoupling of applications than file transfer would since the messages can be transformed before giving the message to the receiver. Also it is much faster since messaging does not require the data to be saved to a folder. Since messaging is fast it can also be used to share applications' functionality. The invoking application sends a request message to the message channel and the receiver replies with processed data as soon as it has finished with processing the request. Unlike with Remote Procedure Calls, the invoking application can still keep on doing other things while it waits for the reply to arrive. [Hohpe & Woolf, 2005] Figure 4 explains how messaging works.

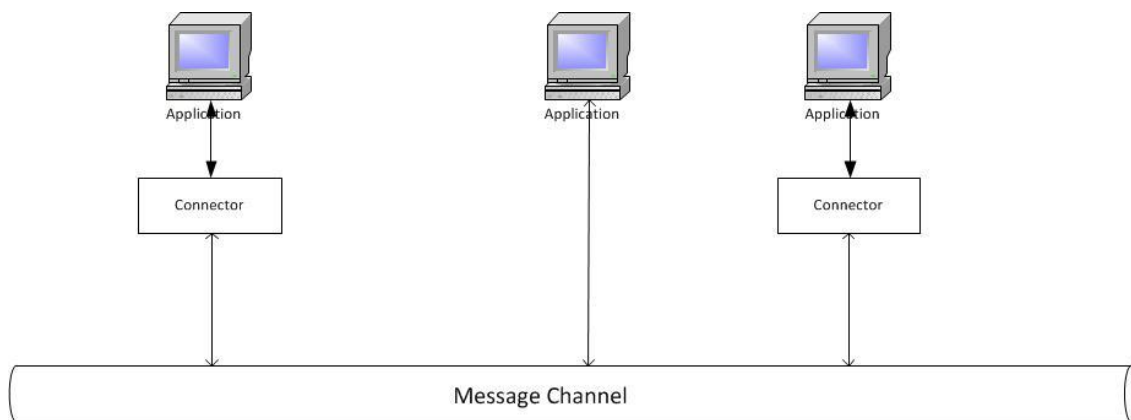


Figure 4: Messaging.

Messaging is not without problems. Because of its asynchronous nature, developing such a system is much more complex than simple file transfer or Remote Procedure Invocation. Unlike shared database it does suffer from inconsistent data between systems since handling messages still takes some time.

2.5. Web Services

Web Services (WS) is one of the latest technologies introduced to the Enterprise Application Integration tools. Its XML [XML, 2012] based communication over common internet protocols such as HTTP [HTTP, 1999], SMTP [SMTP, 2008], FTP [FTP, 2013] and MIME [MIME, 2013] makes connecting applications working on different platforms, programming languages or behind firewalls much easier [Matjaz et al., 2010]. When an application wishes to make a web service call it first needs to connect to the WS server over internet, then send a request, which is an XML based message (SOAP [SOAP, 2007]) or embedded into the URL [URL, 2005] when using HTTP (REST [REST, 2013]), to the interface which then gets interpreted and the WS

Server calls the application from which the data or functionality is needed. The application processes the data and returns an answer to the WS server. WS Server will create an XML message of the result and sends it to the requesting application. Web services follow the Point-to-Point –pattern where applications must be connected through WS interfaces but it enables the applications to be loosely coupled from each other. Also both synchronic and asynchronous communication is possible with web services. In Figure 5 is illustrated how web service calls work.

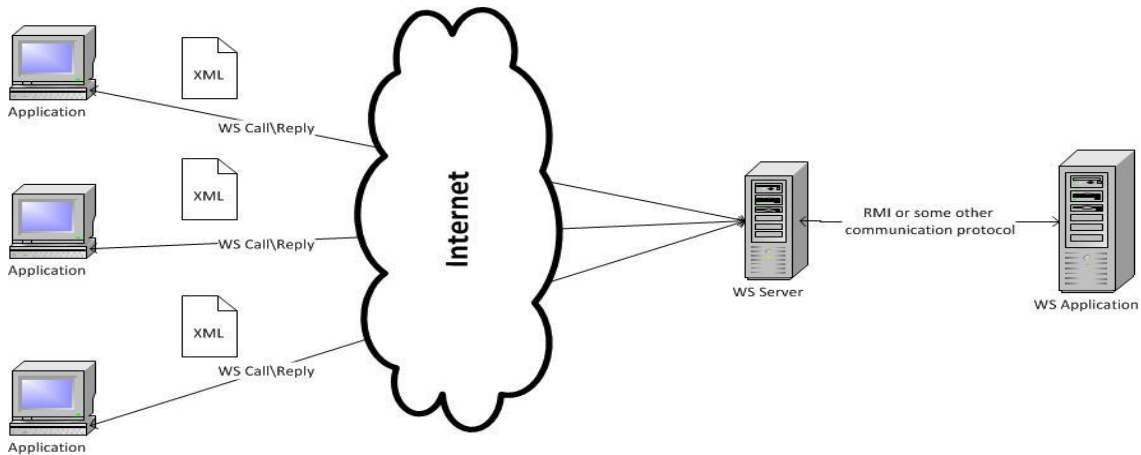


Figure 5: Web Service.

The drawbacks of web services are that the performance of WS calls is much slower than remote procedure calls. Also plain WS does not offer Quality of Service (QoS) [QoS, 1997] features such as security, transactions and other longtime component model features. For these the WS has other additional specifications [Matjaz et al., 2010].

2.6. Point-to-Point

Point-to-Point (PtP) integration is the simplest way of communication between two separate applications. With PtP integration each application is directly connected to all the other applications it needs to communicate with. This also means that there exists a tight coupling between these two applications. This means that both the applications need to be online at the same time for them to work properly. In PtP the integration is quite simple since all the communication happens directly from one application to another. Also the communication is fast as there is no middleware that handles and processes the data before its being received by the other application. The problem with Point-to-Point integration is that although it is good when only a few applications need to be integrated with each other. It will soon start to become too complex when new applications are added to the integration. The more there are applications the more direct connections are needed for the integration. This will soon turn into an integration

spaghetti [Hohpe & Woolf, 2005]. To fully integrate applications together, the amount of integration points for PtP integration architecture with three applications is three for five applications it's already 10 and with 10 applications it would be 45! This is due to the n-rule which means that for n applications an amount of $n(n-1)/2$ integration points is required for full integration [Manouvrier & Menard, 2010]. The situation is illustrated with 3 applications and 2 servers and 10 integration points in Figure 6.

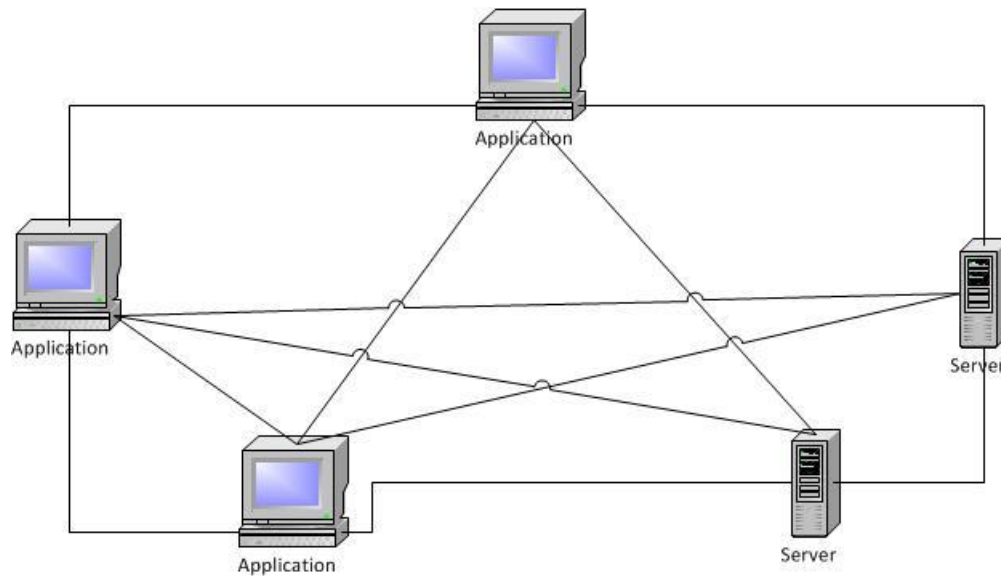


Figure 6: Point-to-Point pattern.

2.7. Broker

Integrating applications using broker style (also called hub and spoke architecture) integration means using a middleware application between all the applications that need to be integrated. The middleware handles the routing and transforming of data for the receiving application. This type of middleware applications are commonly called as integration brokers [Manouvrier & Menard, 2010]. Unlike Point-to-Point architecture where each application has many integration points for each application it needed to integrate. With the broker architecture each application only has one integration point which is connected to the broker. The broker then handles all the message routing and processing before sending the message forward to all the receivers that are interested in the message. This simplifies the integration process greatly when the amount of integrations needed for different applications grow. With broker pattern a single integration point per application is needed. In Figure 7 is illustrated a simple broker architecture

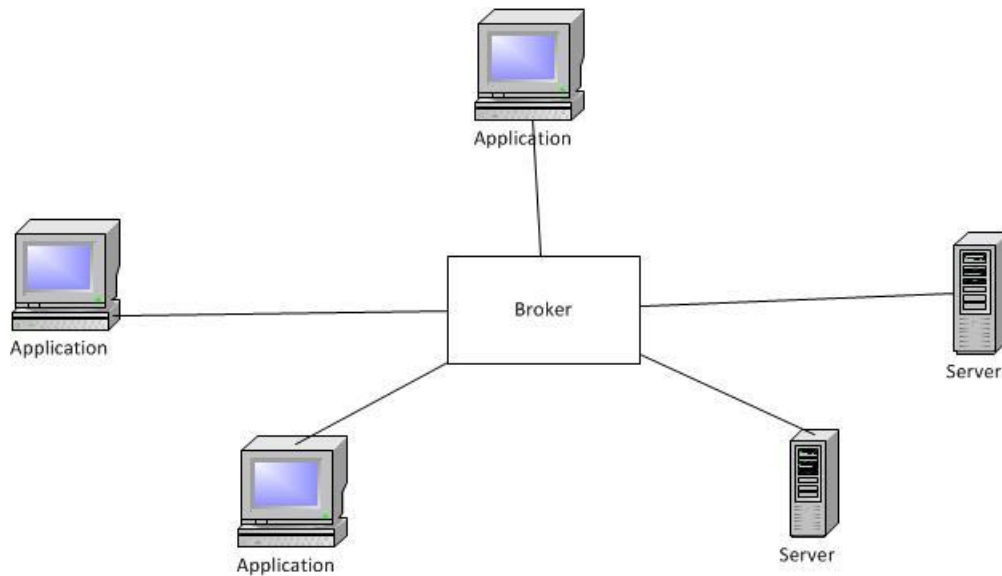


Figure 7: Broker pattern.

There are multiple advantages to broker architecture that makes it an excellent option for EAI. Most implementations of the broker architecture follow the asynchronous communication pattern where connected applications don't need to wait for a response from the receiver but instead can return to doing other work while the broker delivers the messages to correct recipients. The broker architecture also enables loose coupling, of which asynchronous communication is part of, where recipients don't need to be aware of each other or even know whether the other application is currently online. The broker will receive messages from the sending application and route them to the receiving application(s). If one or all of the receivers are unavailable at that time, then it can save the message internally until such time that all the receiving applications have successfully received the message. All this happens without the sending application knowing. Broker architecture also enables a much easier management of integrations since all the configuration is done in a one central location making it easier to modify existing integrations and create new ones.

With the broker architecture the problem is that it creates a single point of failure inside the IT infrastructure. If the platform that supports the hub becomes unavailable or overloaded it can potentially generate a single point of failure (SPOF) [Manouvrier & Menard, 2010]. However this problem can be avoided for example by multiple parallel instances of the broker that work together. If one instance would fail for some reason, the network would still be working since other instances would take the place of the failed broker and start handling incoming messages. [Hohpe & Woolf, 2005]

2.8. Enterprise Service Bus

Enterprise service bus (ESB) is the one of the newest architectural patterns introduced to EAI. The main problem in integration it tries to solve is the broker-pattern's single point of failure vulnerability by dividing the burden into multiple separate components. Salil and Swaminathan [2010] define the ESB architecture like this: *An ESB is essentially an architecture pattern that provides capabilities including connectivity, communications, message queuing, message routing, message transformation, reliable messaging, protocol transformation and binding, event handling, fault handling, message level security, and so on. It enables loose coupling between service consumers and service providers, and thus enhances the ability to rapidly change and introduce new Business Services into the environment* In Figure 8 is an illustration of the architecture of enterprise service bus enabled environment.

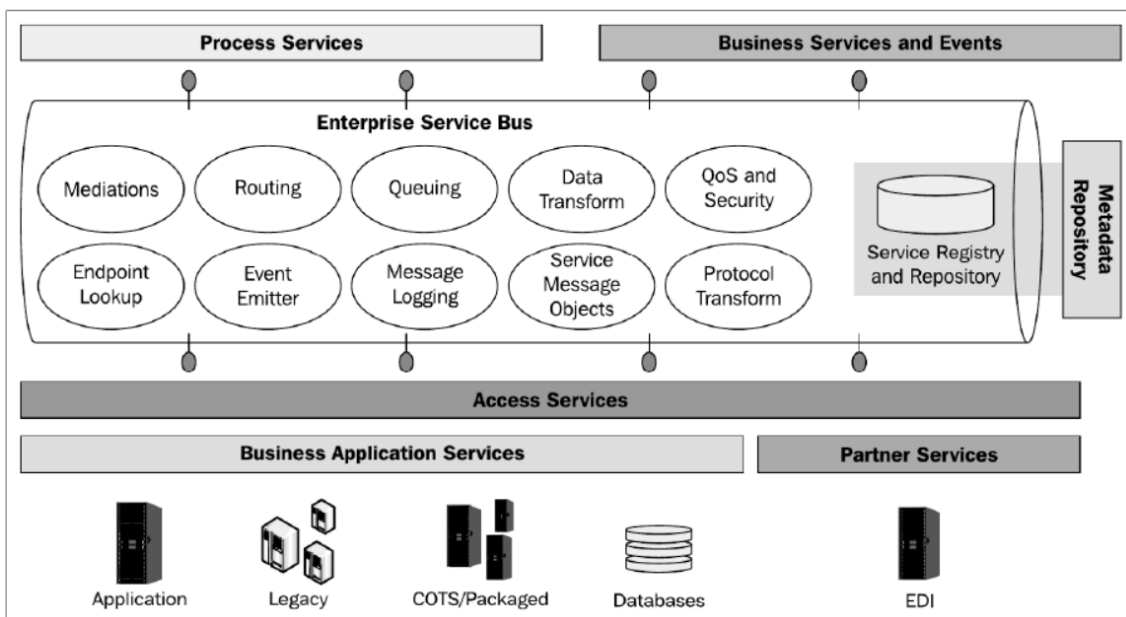


Figure 8: Enterprise Service Bus [Salil & Swaminathan, 2010].

As illustrated in figure 8 the ESB works in the middle of communicating applications mediating messages from sender to the receiver. It receives messages through its endpoints and wrapping them as service message object. Then it routes the messages through different message processor services, data transformers and protocol transformation using message queues and event handling until it can send the message to the receiver through another message endpoint. The ESB can also log the progress of the message during the different steps of the messages. All this is done in a secure way and providing quality of service so that no message is lost.

Although the ESB still relies on a central message bus for delivering messages it divides its functionality to multiple components making the burden of single component

much smaller. With smaller burden the components are capable of working faster and more reliably.

Although being very different compared to each other, There are some core services that most ESBs provide. These are, as defined by Mulesoft [2013e]:

- Location Transparency: A way of centrally configuring endpoints for messages, so that a consumer application does not require information about a message producer in order to receive messages
- Transformation: The ability of the ESB to convert messages into a format that is usable by the consumer application.
- Protocol Conversion: Similar to the transformation requirement, the ESB must be able to accept messages sent in all major protocols, and convert them to the format required by the end consumer.
- Routing: The ability to determine the appropriate end consumer or consumers based on both pre-configured rules and dynamically created requests.
- Enhancement: The ability to retrieve missing data in incoming messages, based on the existing message data, and appends it to the message before delivery to its final destination.
- Monitoring / Administration: The goal of ESB is to make integration a simple task. As such, an ESB must provide an easy method of monitoring the performance of the system, the flow of messages through the ESB architecture, and a simple means of managing the system in order to deliver its proposed value to an infrastructure.
- Security: ESB security involves two main components - making sure the ESB itself handles messages in a fully secure manner, and negotiating between the security assurance systems used by each of the systems that will be integrated.

As any other architectural pattern, the ESB architecture also has its drawbacks. Since the ESB applications architecture is much more complex than for example Point-to-Point -pattern it will require much more initial planning before it can be built and installed. Developers are also required to learn many new concepts that ESB's use.

3. Enterprise Application Integration tools

Enterprise Application Integration tools make integration of multiple applications much easier by decoupling the receiver and sender from each other and handling the delivery of messages to the receivers in the right format. It also simplifies the administration of company's IT architecture since all communication can be handled through a single application instead of each application needing to connect to every other application directly.

In this chapter some of the existing EAI tools are examined. First, some basic information about software quality is provided and then the tools are presented. The tools are organized according to whether they are commercial products or open source implementations. Applications chosen implement the Enterprise Service Bus architecture. ESB tools are one of the newest things in Enterprise Application Integration and so many tools are implemented following that pattern. Although, probably no single integration technology can solve all the problems integration produces [Themistocleous et al., 2004], ESB's seem to be the best viable option for being the EAI tool in the case study. Other implementations seem to be missing some crucial parts or are more concentrated to smaller areas of integration. At the end of the chapter some information of enterprise resource planning (ERP) software is provided. Although not an integration tool as such, the ERP software tries to make integration of separate systems obsolete by trying to include all business functionalities inside one application.

The presented here are only a small part of all the possible integration tools available. As it would have been impossible to go through all the different integration tools on the market inside the scope of this thesis, the amount of tools to be presented was decided to limit to 4 (IBM WebSphere Message Broker, Mule enterprise and community (counted as one), WSO2 and Apache ServiceMix.). There were many other tools that could have been equally viable options for presenting in this paper. Here are some tools that were considered for this paper but were left out:

- Axway B2Bi [Axway, 2013]: Some problems with installation of the tool. Unable to test.
- Apache Synapse [Synapse, 2012]: WSO2 is based on this and is more evolved.
- OpenESB [OpenESB, 2013]: Future uncertain.
- Red Hat JBoss Fuse [Fuse, 2013]: Not enough provided compared to ServiceMix to which it is based on.
- JBoss ESB [JBoss, 2013]: Not as evolved as others.

Each of the chosen tools was installed on a Lenovo W500 laptop for the purpose of research and testing. Also more information about the tool was gathered by researching the internet.

3.1. Software Quality

The quality of software is a crucial factor when deciding on acquiring an application. As the criticality of software increases so does the importance of the software's quality. Unreliable software being used in a highly critical position, like monitoring safety of a nuclear plant, will most certainly increase the possibility for disaster. Problem is how can it be known whether an application is of good quality before acquiring it? To verify the quality of software it needs to be defined what are the precise requirements for the software. [Hughes & Cotterell, 2009]

The international standard ISO/IEC 9126 for the evaluation of software quality defines a standard approach for evaluating the software quality in a way that removes some well-known human biases from the process. The standard consists of four different parts: quality model, external metrics, internal metrics and quality in use metrics [ISO/IEC 9126, 1991].

The quality model specifies a set of characteristics of quality software. These characteristics are Functionality, Reliability, Usability, Efficiency, Maintainability and Portability. Functionality specifies the functions that fulfill a set of implied or stated needs, Reliability defines the level of performance for certain period that the software is required to be capable of, Usability specify the needed effort and experience by users from using the software, Efficiency defines how effectively can software use the resources that are allocated to it in terms of performance, Maintainability specifies the required effort to maintain and modify the software and Portability defines how easy the software is to transfer from one environment to another. Each of these characteristics is further divided into multiple sub-characteristics. The sub-characteristics are listed below [ISO/IEC 9126, 1991]:

- Functionality
 - Suitability
 - Accuracy
 - Interoperability
 - Security
 - Functionality Compliance
- Reliability

- Maturity
- Fault Tolerance
- Recoverability
- Reliability Compliance
- Usability
 - Understandability
 - Learnability
 - Operability
 - Attractiveness
 - Usability Compliance
- Efficiency
 - Time Behaviour
 - Resource Utilisation
 - Efficiency Compliance
- Maintainability
 - Analyzability
 - Changeability
 - Stability
 - Testability
 - Maintainability Compliance
- Portability
 - Adaptability
 - Installability
 - Co-Existence
 - Replaceability
 - Portability Compliance

The internal metrics are specified as those that are always static regardless of any software execution related variables. Internal metrics are mostly relevant when developing software and not in cases when choosing from existing products. The external metrics specify the metrics related to running the software on different setups and platforms. Quality in use metrics defines the metrics that are only applicable in real use situations and not otherwise testable. In ideal situations the internal quality should determine the external quality and quality in use should be determined by the external quality. [ISO/IEC 9126, 1991]

The importance of each characteristic and sub-characteristic should be judged for each application separately. Reliability is very important in monitoring nuclear plants as efficiency is important for software working under strict time constraints. For each

quality there should be an external measurement defined which verifies that the said quality is found in software. The ISO/IEC 9126 specification gives a standard list of quality characteristics but under the characteristics and sub-characteristics are the actual requirements that the software in question should adhere to. [Hughes & Cotterell, 2009]

To be able to decide whether a system meets its requirements there should be a way to measure its qualities [Hughes & Cotterell, 2009]. The requirements should be stated in a manner that it is possible to decide from objective measurements whether the software fulfills it. A requirement that cannot be objectively measured can easily become a subject of biased inspection where a person's views can affect the decision of whether the requirement is fulfilled or not.

Collecting the requirements for software can be done in many ways. Requirements can be collected doing interviews, doing research on existing similar software implementations built for similar purpose as the target software, collecting information about the business domain that the software is supposed to be working in or making case studies of the ways the software is operated and how the users are using it. These are but few ways and there are more possible ways of collecting the requirements. As using only one collection method will not be sufficient to locate all the requirements, it is useful to include more than one collection method.

3.2. Commercial EAI Tools

In this part of the chapter we look into some of the commercial EAI tools available. The tools were chosen by doing research using the internet and choosing the Enterprise Application Integration applications that were found to be probably the most suitable.

3.2.1. IBM WebSphere Message Broker

IBM WebSphere Message Broker is an enterprise service bus (ESB) providing connectivity and universal data transformation for service-oriented architecture (SOA) [SOA, 2013] and non-SOA environments. [IBM, 2011]

IBM WebSphere Message Broker works on top of WebSphere MQ, a Message Oriented Middleware (MOM) [Curry, 2004] implementation also from IBM, and requires it to be installed on the same machine. The WebSphere MQ is included in the licence fee and incurs no extra costs. IBM WebSphere Message Broker also requires the Message Broker toolkit to be installed which is a graphical user interface for managing the message broker and its messaging flows.

The tool is capable of dynamically mediating interaction between services and routing and transforming message data. It supports Microsoft .NET environments including CLR [CLR, 2013]. It can be scaled to multiple servers. It has a browser based diagnostic tool and the data-mapper tool is based on open source and built on top of Eclipse. Figure 9 illustrates the Eclipse based data mapper tool.

IBM WebSphere Message Broker supports multiple protocols including WebSphere MQ, JMS 1.1 [JMS, 2002], HTTP and HTTPS, Web Services (SOAP and REST), File and TCP/IP [TCP/IP, 2013]. It also supports multiple data formats including binary formats (C and COBOL), XML, industry standards (including SWIFT [SWIFT, 2013] and EDI [EDI, 2013]) and user can create his/her own custom formats. [IBM, 2011]

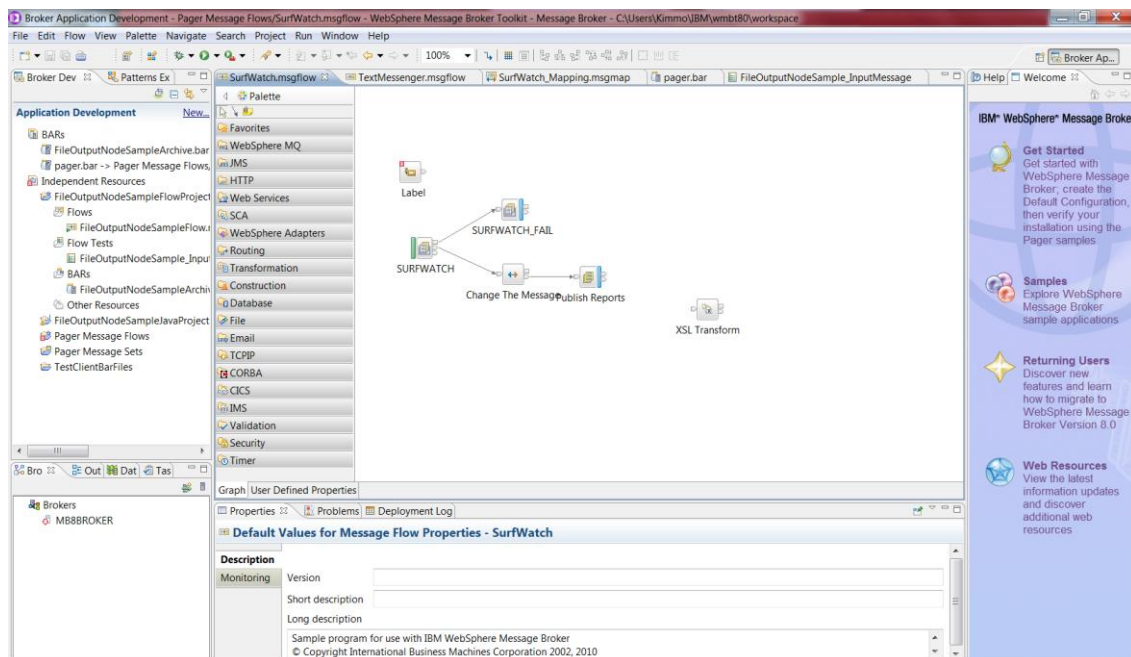


Figure 9: IBM WebSphere Message Broker.

The IBM WebSphere Message Broker seems quite monolithic with its almost 2Gb combined size installation packages compared to just few hundred megabytes in most of the other tools that were looked into. Although IBM's own feature list advertises the WebSphere being easy to use and install this was not the experience when trying to install and use it. The graphical user interface of the toolkit seemed quite simple but trying to use it proved to be not so simple. For some reason the examples did not work although everything was supposed to be configured correctly. This is something that is not expected from a commercial tool. The initial impression of complexity can easily turn users away from the product just because they can't even get the examples working.

3.2.2. Mule ESB Enterprise

Mule ESB is an integration tool developed by MuleSoft [2010]. It was first released in 2003 and was one of the first open source ESB's in the market [Cope, 2010]. Current stable version is 3.3.2.

There are two versions of Mule ESB, the Mule enterprise and the Mule community. Mule enterprise is a commercial product with more features to provide better reliability, security, performance, control and message transformations. It includes a graphical web based management console. A customer that buys the enterprise license also gets better access to different support channels from MuleSoft. The community version of Mule ESB has fewer features and lacks the graphical management console but is completely free to use. In this subsection we concentrate on the enterprise version of Mule ESB. In the Free Open Source part of this chapter we will briefly look into the Mule Community. Nevertheless large part of what is told about Mule ESB enterprise will be true about the community version also.

Mule ESB has an open source core, uses open standards and is based on Java programming language. It's lightweight, flexible and scalable both vertically and horizontally. Horizontal scaling means adding more nodes (For example servers) to the system and vertical scaling is done by adding more resources to the existing node(s) (for example increasing memory or CPUs to the server). It's also promised to be capable of 100% uptime and is vendor-neutral so different vendor implementations of app engines can be used to run it. [MuleSoft, 2013b]

Mule ESB has a wide range of connectors and transports (CXF [CFX, 2013], email, FTP, Hibernate, HTTP/S, JMS, LDAP [LDAP, 2006], RMI, SOAP, TCP, etc.). It has a strong support for Web Services for multiple frameworks and protocols (Axis [Axis, 2010], Atom [Atom, 2005], CXF, .NET Web Services, REST, etc.). Message transformations can be done in multiple different mechanisms (XSLT [XSLT, 1999], XQuery [XQuery, 2010], Smooks [Smooks, 2012], Oakland (not open source) [Oakland, 2013]). [Cope, 2010]

The Mule ESB has, roughly said, two parts, the connectors\transports that handle the delivering of messages from the sender to the receiver and the integration services that handle the routing, transaction management, security, transformation, transportation management and message brokering. Figure 10 illustrates the components of Mule ESB.

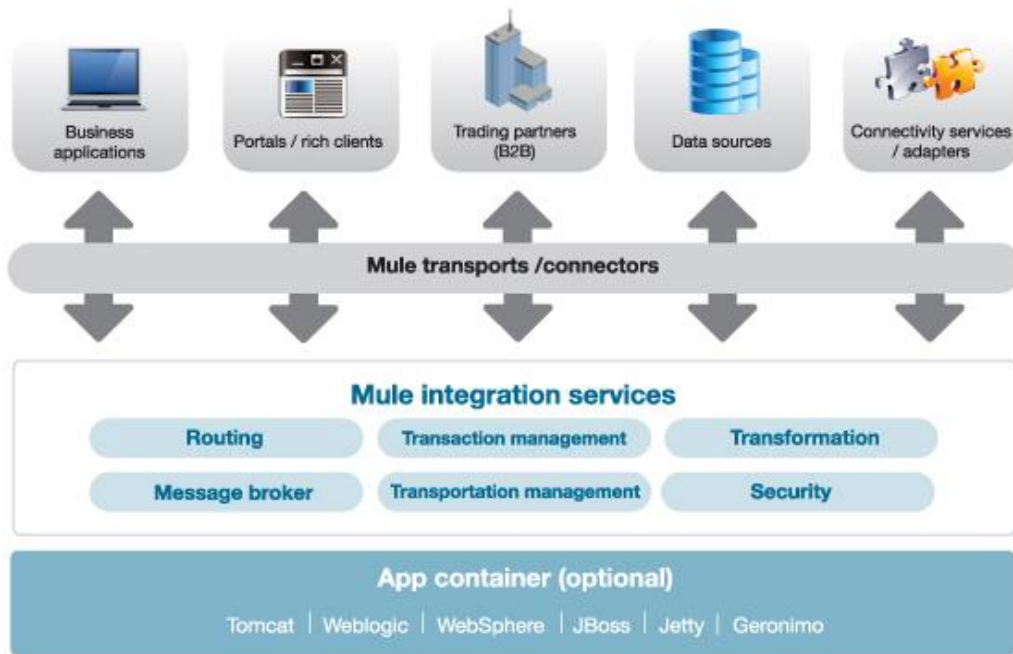


Figure 10: Mule ESB [MuleSoft, 2013c].

In addition to the Mule ESB, Mulesoft offers an Eclipse [Eclipse, 2013] based graphical design environment called MuleStudio for creating different integration applications which can then be easily deployed to the Mule ESB. One of the most important features of the MuleStudio is the DataMapper. With it the developer has much more detailed access to the data inside the messages and can process it in any way he/she likes. The DataMapper feature is only available in enterprise version of Mule ESB. Figure 11 shows the graphical user interface of MuleStudio.

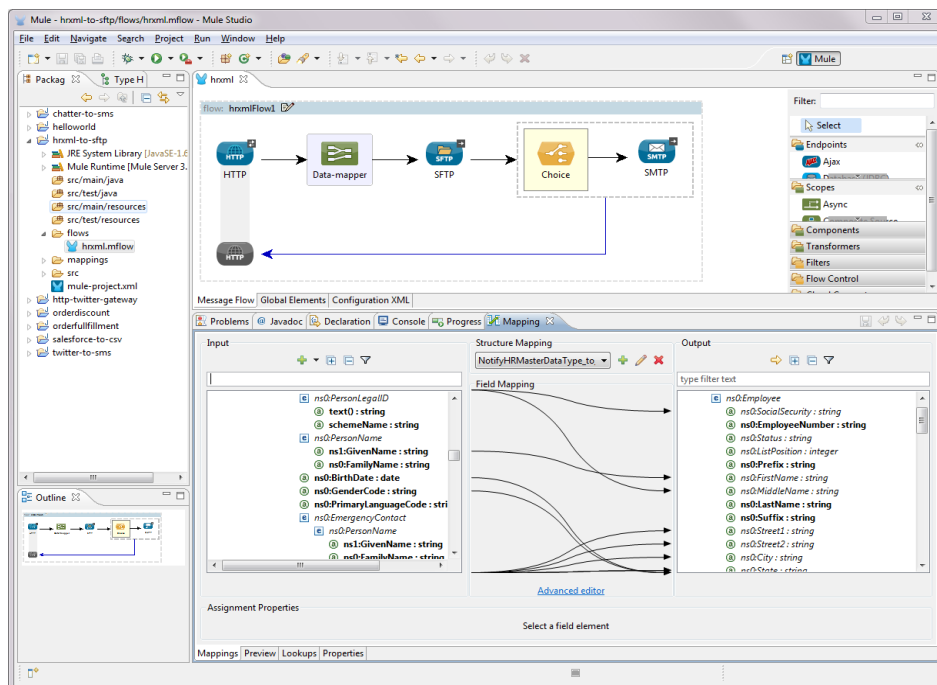


Figure 11: Mule Studio [Burton, 2012].

Although the MuleStudio seems quite simple and easy to use, the initial experience of the tool made a very different impression. Many of the components seemed to have lots of different configurable boxes without much information on what should be put into them. Although wide configuration options were seen as a good thing it was thought that the implementation left much to hope for. Luckily the documentation and tutorials for MuleStudio are quite good so with some patience it was relatively simple to get comfortable with the tool. Still it must be said that there is some learning curve to getting familiar with all the components and using them efficiently. The DataMapper tool seemed to be quite an efficient tool for message transformations and the scripting language used for it seemed easy to learn.

Another component only available to the enterprise version is the web based management console. The management console provides a complete view to all instances of Mule ESB installed across the enterprise. From the management console a developer can view servers, clusters, applications, flows, alerts and errors; manage applications, deployments and server resources and perform administrative tasks [MuleSoft, 2013d]. Basically everything required to run and configure the Mule ESB should be manageable from the management console. Figure 12 illustrates the graphical user interface of the management console.

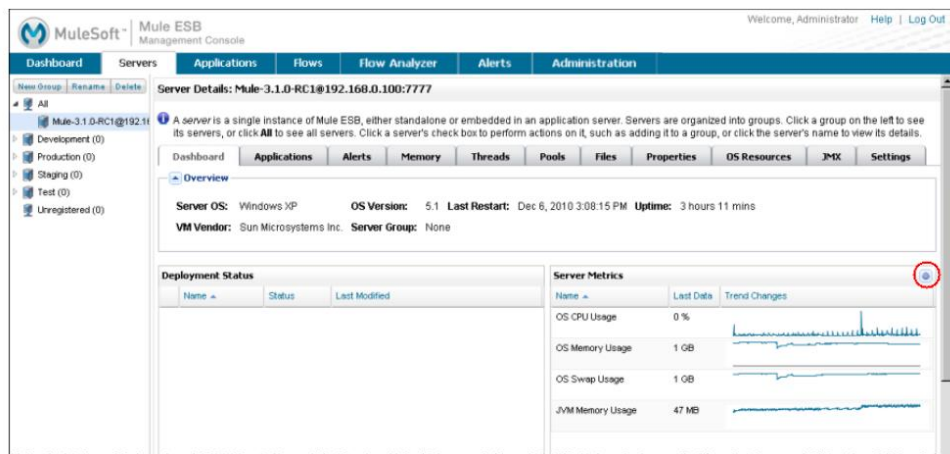


Figure 12: Mule Enterprise GUI [MuleSoft, 2013d].

As with the MuleStudio, the Management console also gave an initial impression of having lots of different ways to manage Mule ESB but also was a bit confusing to use and there certainly is a learning curve to using the Management console. A nice feature is the Flow Analyser that allows users to analyse different flows separately providing better visibility to the problematic flow and what is happening inside it.

3.3. Free Open Source EAI Tools

In this part of the chapter we look into some of the free open source EAI tools available. These tools were chosen using the same methods as were used in the commercial tools.

3.3.1. WSO2 ESB

WSO2 is a company started in 2005 and released their first Enterprise Service Bus in 2007 and current version of the ESB is 4.6.0. It's built on Apache Synapse ESB. It uses ActiveMQ for message queuing and Axis2 for web services. It is concentrated on using web services as message processing components. WSO2 ESB [WSO2, 2013] is licensed under the Apache 2.0 License. There is no commercial version and it is 100% open source. [Cope, 2010]

WSO2 supports multiple common transport protocols (HTTP, HTTPS, FTP, JMS, etc.). For message transformation it uses java framework called Smooks [Smooks, 2012]. The web management console includes an interface for creating transformations but it does not remove the need for coding [Cope, 2010].

WSO2 has a surprisingly good looking and extensive web based management console which is not so common on open source tools. Inside the management console is basically included everything from message routing and transformation to server management and configuration. Figure 13 shows the homepage of the web console.



Figure 13: WSO2 web console.

Configuring the WSO2 Enterprise Service Bus is somewhat difficult and requires some studying and work just to get the samples working. Luckily the community around WSO2 ESB seems quite active and there is extensive documentation on how to get started.

3.3.2. Apache ServiceMix

Apache ServiceMix is a free open source implementation of an Enterprise Service Bus licensed under Apache license v2. ServiceMix project started in 2005 and is now in version 4.5.1. It can be run as a standalone or in an application server.

ServiceMix supports multiple application server vendors from which the user can freely choose one. Initially ServiceMix was based on JBI [JSR208, 2005] but has since migrated to OSGi [OSGi, 2013] though it still supports JBI. There is a commercial Enterprise Service Bus implementation called Fuse ESB from RedHat that is heavily based on ServiceMix and can be thought of as the commercial version of ServiceMix. [Cope, 2010]

ServiceMix introduces a modular approach to Service Oriented Architecture (SOA). It uses Apache ActiveMQ [ActiveMQ, 2013] for messaging, Apache Camel [Camel, 2013] for Enterprise Integration Patterns and routing messages and Apache CXF for web service connections. It has an OSGi-based runtime by Apache Karaf [Karaf, 2013] and a complete WS-BPEL [WS-BPEL, 2007] engine with Apache ODE [ODE, 2013]. All the components are loosely coupled using Apache ServiceMix NMR. [ServiceMix, 2013] Figure 14 shows the overview of the architecture of ServiceMix.

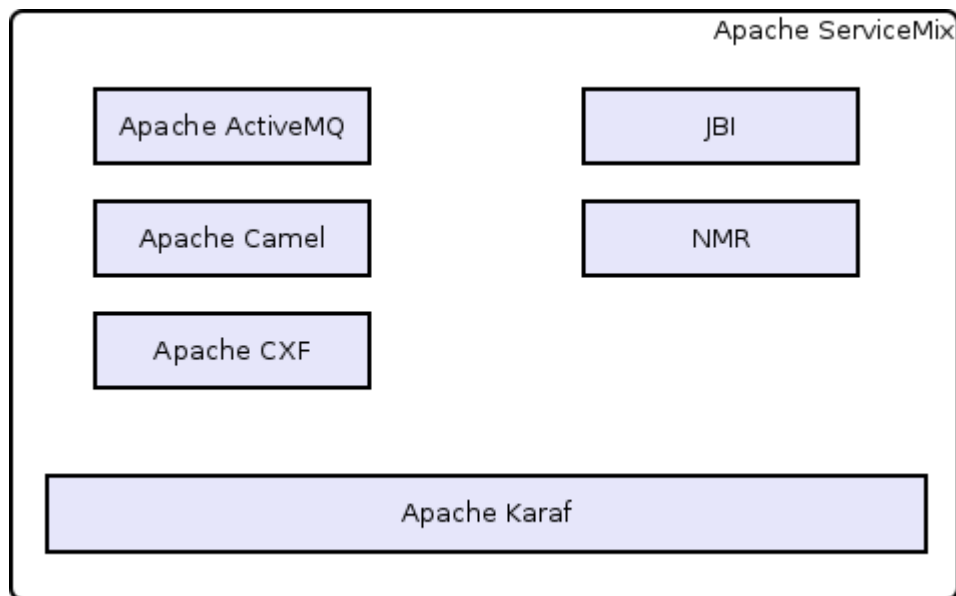


Figure 14: ServiceMix overview [ServiceMix, 2013].

ServiceMix's Apache Karaf based management interface is on command console but a web based management console is also available for install as a feature. The web console is quite basic and the command console is actually also embedded into the web

console to provide all the same functionality inside the web console as is in the command console.

There is no actual data mapping tool for creating message transformations with ServiceMix but it supports multiple ways of creating your own transformations including XSLT, JSR223 standard based scripting engines [JSR223, 2006], Web Services and JAXB [JSR222, 2009]. In the latest version the JBI modules are still supported but not recommended and CXF is recommended instead. ServiceMix also supports a wide range of connectors including File, FTP, HTTP, JMS, Email and SOAP [Cope, 2010].

3.3.3. Mule ESB Community

Mule ESB Community Edition is a stripped down version of the commercial version missing some important features such as enterprise hardening with back ported bug fixes and the DataMapper-tool. Also the management console has been left out and monitoring of the community version requires using java's JMX-framework [JMX, 2006]. The JMX management tool does not support all the features provided in the web-based management tool. These features include operational dashboard, deployment manager, integrated application repository, ESB remote control, task scheduler and the REST API for management. The community version also lacks some special proprietary connectors like SAP and IBM WebSphere connectors. As the community version does not have the Mule Enterprise Security package which is included in the commercial version it is not as secure. Performance and reliability of the community version has also been greatly reduced by omitting the high availability and caching features. Also there is less options for performance management than what is included in the commercial version. The commercial version includes features like SLA [Bouman et al., 1999] alerts, monitoring framework integration, runtime performance manager and key performance indicator tracking. Also many analysis tools have been omitted that would help problem resolution and the support is only limited to the community forums. Lack of so many features basically makes the community mule somewhat a crippled one.

Unfortunately, as listed above, all the most important features are inside the commercial version and it seems that MuleSoft is pushing the commercial version to customers quite heavily so the open source version is left missing on some important features. Deciding to use this tool would require commitment on taking the commercial route. [Cope, 2010]

3.4. Enterprise Resource Planning

ERP (Enterprise Resource Planning) system has a major role in a company, its purpose is to integrate the company's different facets of operation into a one big information system and provide up-to-date information in real time to all interested parties [Patrick, 2002]. According to Malhotraa & Temponi [2010] *ERP is an enabler for technological integration that has evolved from basic Material Requirement Systems (MRP, MRP II) to sophisticated and multimillion-dollar systems that aim to link databases and applications in a friendly manner.* One of the ideas behind ERP systems is that it could answer to all business needs a company might require. This way all the business logic would be inside one system making the integration of multiple applications obsolete. Most of the time this, of course, is not the case in real life.

Normally the ERP systems take quite a large and central part of the company's IT architecture and most of the business information is stored inside its data storages. ERP's are quite expensive to acquire and require much training for the staff of the company. The company might be even required to change some of its procedures and processes along the way to better utilize the ERP system and its procedures. Many times, due to the extent of the system, the job of implementation and maintenance are outsourced to an expert organization. For these reasons the ERP system is also normally the last thing to be changed inside the company. [Rinneheimo, 2008]

Often when a company purchases its first ERP system there is already a working IT architecture inside the company built using multiple different applications. Sometimes the old applications might be better than what the ERP system can provide as a replacement. If the company decides that they should be able to use some of the old applications even when the ERP system is in use, those tools need to be integrated with the ERP. Most companies also don't work without any connection to the outside world but have connections to outsider organizations that they must communicate with, sending billing information to customers for example. Whether it's integrating an in-house application or communicating with an outsider organization, integration from the ERP can happen mainly in two ways: Either the ERP itself has a built in system that provides connections to other systems or some kind of external integration system is required. The external integration can range from running stored procedures that extract data from database into files to using a professional EAI tool that handles all the integration needs between the ERP and other parties. Figure 15 shows some of the different business sections that can be included inside an ERP.

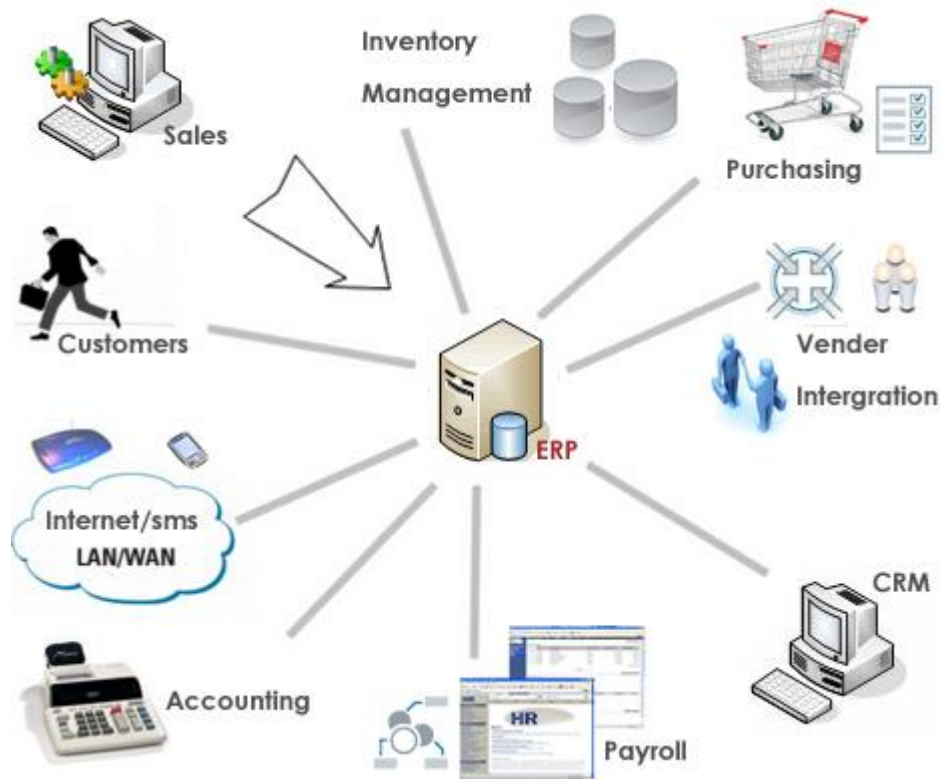


Figure 15: ERP business sections [Acsonnet, 2013].

4. Case Study: Glass & Frame

4.1. Merit Consulting Finland

Merit Consulting is an IT consulting company selling and supporting installation and implementation projects of Enterprise Resource Planning systems (ERP). Merit also provides support and maintenance services for users of ERP systems who are already in production. They are mainly focused in customers located in the area of Northern Europe. The ERP's come from a third party vendor that is responsible of the development of these systems and Merit is a licensed reseller of these products. Their own products are mainly "value added"-products which support the customers usage of the ERP's that the company is providing. This study concentrates on their Finnish branch Merit Consulting Finland.

A couple of years ago Merit Consulting Finland (Merit FI) added a second ERP called Jeeves to their selection. This new ERP focused on smaller customers than what their first ERP was capable of reasonably supporting. With the new ERP they would be better equipped to expand their customer base in Finland which is a small marketplace for the kind of ERP's that their initial system represented. Unlike the old ERP that came with its own tools for integration, Jeeves didn't really have any good integration tools. Merit FI was faced with a new problem. How can they easily integrate Jeeves with the customer's existing IT architecture?

The author of this research paper is also an employee of Merit FI working currently for the fourth year as an integration consultant. The domain experience of the author was utilized during the requirements specification process.

4.2. Jeeves

Jeeves is a Swedish ERP company that provides enterprise resource planning solutions for small and medium sized companies (SME). They have two ERP solutions that work on the same platform but are targeted for different customers. Jeeves Selected for smaller businesses and Jeeves Universal for medium sized businesses. The companies using Jeeves's ERP can vary from 5 – 1000 employees and can be from three different branches of industry. The three branches are Distribution, Manufacturing and Service industry. The idea of having two products on the same platform is that when the company using Jeeves ERP grows from small to medium sized it is very easy to expand the existing ERP to provide for the growing needs of the company. [Jeeves, 2013]

The selling points of Jeeves ERP are flexibility, upgradeability and ease of use. ERPs are quite costly and time consuming systems to implement into the company and the ERP will be a major part of the company's business for many years to come. As such, it is very important that the ERP system can be easily modified to best serve the business needs of the company. Jeeves ERP has been built in a way that each building block can be extended or have some functionality added to it. For adding new apps Jeeves has developed an apps technology to enable easy addition of value-add customizations and applications. For anybody who is familiar with ERP's, the process of upgrading one is seen as a project with weeks or months of work and stress especially with installing customizations to the new environment or maybe even rebuilding them. Jeeves has been built so that all adaptations are installed to a separate database so that upgrades won't affect them. As a result 80% of Jeeves customers have the latest version of Jeeves ERP in their production use. [Jeeves, 2013]

Jeeves ERP is based on Windows platform and uses Windows SQL server 2008. It is only compatible with the Windows SQL server and does not support any other Database. This is due to the fact that all business processes of Jeeves are built as SQL applications or stored procedures of which users can connect to using a client application. Since Jeeves is SQL based application, integration to it will need to happen through connecting to the database and using SQL.

Making straightforward SQL queries to a commercial enterprise product is not very safe in terms of data integrity and security so Jeeves has developed a product called Jeeves Service Builder (JSB). JSB is basically a Web Service creation tool that provides a level of abstraction between the third party application and Jeeves ERP. It enables developers to create web services that handle requests from a third party to Jeeves using custom stored procedures, SQL queries or Jeeves's own applications inside the database. With the JSB tool a developer can create chains of events that will be executed when the Web Service request is made. Unfortunately the tool is somewhat difficult to use and as such can easily drive away potential users to using other solutions. Also a Web Service is only good if the other party is actively requesting info. What if you need to send data without someone requesting it or request data from another? For this you would need some other tool.

4.3. The Customer

As the tool is required to be used by many different customers on different branches of industry it was thought that best course of action would be to artificially create a nonexisting example customer with needs that would match the different integration needs of multiple different customers. The different integration problems have all been

extracted from previous work with multiple customers and are presented together as the needs of the customer.

The customer is a company called Glass & Frame Ltd. From now on we call the customer as G&F. G&F is working in the manufacturing industry making windows and window frames for buildings from summer cottages to apartment buildings and from government offices to shopping malls. They also make custom window frames when their customers cannot find what they want from the normal product line. They deliver their products all over northern Europe but are also planning on expanding to Russia. G&F has just purchased Jeeves ERP from Merit FI to replace their in-house developed ERP system which is quite old and there are no more developers in the company to improve it.

The In-House ERP did not have any specific tool for creating integrations. Instead any new integration point that was required to introduce new applications or features to the ERP system needed to be coded straight into the ERP's source code. This was not a very efficient way and as the original developers slowly left the company there was no one to make the changes anymore. At this point G&F started to search for a commercial ERP product to prevent this from happening anymore.

4.4. G&F integration needs

G&F has thousands of customers around northern Europe, some companies and some private persons, who use four different communication lines to place orders for windows and frames. The customer can go to a website that G&F is co-hosting with many other construction related companies from where they can place an order for windows or frames while also ordering other items needed for building a house or the customer can go to the G&F's own website and places an order from there. The customer can also go to the local hardware store and order the windows there if the hardware store owner has made a contract with G&F. In this case the hardware store worker will input the order in to the G&F's website. The last option, which is used by many larger companies with intention to order larger amounts, is to call directly to one of the salespersons of G&F and the salesperson will make the order for them.

Although there are four ways of making the order, there actually is only two ways that the order information comes to the ERP system to be handled. If the order is placed by talking with one of the salespersons, the salesperson will place the order straight into the ERP system and no integration interface is needed. For all two (as hardware stores also use the G&F website) of the other communication lines there is a unified interface

that is used by both websites to send in their orders. The unified interface is XML based and currently works so that the websites send an XML file to a FTP-folder from which the ERP reads and processes the file.

G&F also needs to send invoices to customers. This happens by G&F sending the invoices in XML format to their bank and the bank will convert the XML messages into invoices and send them to customers. There is also need for accounting interfaces with three different banks which are handled using EDI-format [EDI, 2013]. All of these messages are sent to the banks using FTP-protocol and the banks send their messages similarly using FTP-protocol back to the ERP. Some of the order- or invoice messages grow to be quite large and might be even 20MB in size.

Lastly G&F's ERP needs to communicate with two different warehouse management systems. The communication between the systems is currently handled using FTP transfer as is with the banking interfaces. However as the message traffic during daytime is quite high with these two interfaces, up to 40 000 messages a day, sometimes this solution with the old ERP is too slow and some there starts to be a queue of messages waiting to be processed in the FTP server. This is due to the fact that the ERP only picks up a new message after it has processed the previous one and there is only possibility for handling one message at a time or otherwise the stock quantities would get unsynchronized with the external system. As these messages are time critical and should be handled within seconds from creation the situation is more than just undesirable. The warehouse systems provide interface for JMS-queue connection but the ERP has no such connection options.

Currently there are also some nightly batch runs to synchronize databases between the external warehouse management systems and the ERP and creating different reports. G&F hopes the reports could be run anytime but this is not possible with the current ERP system as the ERP is struggling with the other warehouse messages.

The old ERP handled, with varying success, almost all of the business needs that were required by G&F. It could send invoices, create orders, handle accounting, manage warehouse stock inside their small local warehouse for test and custom products as well as communicate with the larger external warehouse management systems that they used for warehousing their main product lines. Now with the new ERP they hope to be able to at least do the same but more efficiently and also possibly introduce a new warehouse management system to their internal network.

There are some integration tools that Jeeves offers as third party solutions, such as Tacton [Tacton, 2013] and StreamServe [StreamServer, 2013], but problem with them is that they concentrate only on one or two things making the integration scatter to multiple applications. This could easily mean big costs to customers with extensive integration needs. Also it would be much more difficult to maintenance multiple integration tools rather than just one. Since there was no prior experience with these tools and none of them seemed to be able to cover a large enough area of integration it was decided by Merit that they should find another tool that would fit their own and their customers' needs better.

4.5. Requirements

Since Merit Consulting is not the end user of the ERP solution but a consultant company that sells the ERP license to its customers it has to carefully think about the requirements of the EAI product in the perspective of all of its possible future customers. Fortunately Jeeves being the second ERP product instead of first they already have years of experience with the integration needs of customers from working with the first ERP system. The requirements for the new tool were identified by using this experience, interviewing an employee of Merit FI and also looking into some of the existing customers' integration solutions in the form of the case study explained earlier.

The employee interviewed is currently working at sales department of the company but started and worked many years as integration consultant and as integration team leader in the company and has also been heavily involved in the current Jeeves projects. The interviews were conducted in a free style without a specific structure. It was seen that in this way the conversation would be more relaxed and there would emerge more requirements than from using a strict structured interview style where the person interviewed would be more concentrated on answering questions than finding out the needs for their customers. During the interview the interviewer's job was to steer the conversation back to the topic by asking questions whenever the conversation seemed to be slipping out of the topic. Requirements from the interviews were extracted by using the interviewee's experience in the domain and reading through the notes written during the interview.

Although being architecturally very different from Jeeves and having an integration product of its own, the different integration patterns that were needed were considered to be very much the same with Jeeves as have been with the first ERP. It was seen that integration challenges in smaller companies did not introduce any additional integration challenges than what Merit FI was used to handling with the larger companies. Only challenge that was identified with smaller company size had to do with selling the EAI

tool to Jeeves ERP's customers since it was assumed that in their view it would introduce extra costs that they might not see necessary or could afford to spend. After all it was possible, although more difficult, to implement the needed integrations without an EAI tool. Since the companies would have much tighter budgets than bigger companies, any tool they would buy would need to be a low costing product but still capable of handling all of their integration needs.

Based on the example customer and interviews with Merit personnel nine main branches of requirements for the EAI tool were identified as:

- Stability (Maintainability)
- Changeability (Maintainability)
- Security (Functionality)
- Performance/Scalability (Efficiency)
- Connectivity (Functionality)
- Management/Monitoring (Maintainability)
- Message Transformation (Functionality)
- Routing (Functionality)
- Pricing (N/A)

The list above is not in any specific order. Next each of the topics is explained in more detail and requirements related to it will be listed in the order of importance. After each requirement is marked the branch to which it belongs and the importance of it to the EAI tool. The importance of the requirement is expressed in numbers ranging from 1 to 10 where 1 means the most important and 10 is the least important. The scale was chosen on the basis that it was found the most suitable one. The scale indicates how important Merit FI saw each requirement and the numeric values should not be seen as general level of importance for the requirement.

It should be noted that requirements derived from G&F are actually a combined collection of problems and situations that have happened in past and current projects with Merit FI. G&F used in this paper is not an actual company.

Example requirement:

- *The tool keeps all the authentication credentials protected from unauthorized access. (5)*

4.5.1. Stability

As with any mission critical application, the EAI tool must be stable enough to work online for long periods of time without errors or requiring any maintenance that would

stop the tool for a period of time. In the case of G&F research revealed that the application was only allowed to be down during weekends meaning a minimum of 6 days without maintenance to the system was required.

When errors occur it shouldn't affect any other component than the one that caused the error. Recovering from such errors should also be as automated as possible. If a single message processing error could crash the whole system it would be a disaster.

- The tool saves data of errors for later inspection. (1)
- The tool is capable of withstanding errors situations of its components without crashing completely. (1)
- The tool can alert of any error situations that might occur. (2)
- The tool is capable of automatically recover from most error situations. (4)
- The tool is able to work continuously for 6 days per week. (5)

4.5.2. Changeability

Inside Merit FI it was understood that no EAI tool could possibly be able to answer all of the requirements the customers might have for it. So it was agreed that the EAI tool should provide interfaces for easy customizing and adding of features by Merit's own employees instead of always contacting the vendor and requesting for new features. By being able to customize the tool Merit FI felt it would be better equipped to serve its customer's needs faster and more accurately.

- The tool provides an interface for adding new connection options. (2)
- The tool provides an interface for adding new transformation options. (2)
- The tool provides interfaces for adding features and customizations to the tool. (3)
- The tool provides documentation on how to add features and customizations to the tool. (5)
- The interfaces for customizations are simple to use. (6)
- The tool's source code is available for inspection. (7)
- The tool's source code can be modified by Merit FI. (8)

4.5.3. Security

Security of the EAI tool was divided into two categories: Security of the application against harmful attacks and message transaction security. Message transaction security meaning how certain can we be that the message that is sent through the tool will be received by the right receiver.

G&F has had some incidents in the past where a hacker has tried to attack their systems. For this reason they require that the tool is secure from any attacks. Most of the time the tool works inside the company network with only a limited access to the outside world so the many of the security issues fall for the customer and not a lot can be done by the application itself. In situations where an attacker has gained access to the customers servers and is searching for ways to do harm the EAI tool should not provide an easy access for this. For example, all usernames and passwords should be hidden and preferably encoded instead of residing inside text files in plain text. An access to one part of the system can easily provide accesses to other parts of the system. Also if the attacker can start following or changing the messages going through the EAI tool they can easily cause large monetary losses to the customers.

The old system at G&F lost information sometimes which caused a lot of excess work to locate the lost data or calling the customers to re-create their orders. Message transaction security is very important issue since a loss of a single message can cause hours of labor and cost thousands of dollars for the customer. Thus any message that is being processed by the EAI tool must also be delivered to the right receivers and only the right receivers. Even in situations where the server crashes and goes offline. When the server is back online again it should be possible to continue the processing of the messages that were being processed before the crash. Whether this is automated or done manually does not really matter although automated reprocessing in some cases would be preferable.

- The tool makes certain, that any message received by the tool will get delivered to the receiver(s). (1)
- Only the correct receivers will receive the message. (1)
- The tool saves messages in case of system crashes so that it can return processing after the tool becomes online again. (1)
- Access to monitoring tools is password protected. (2)
- The tool can use commonly known secure connection protocols (HTTPS, FTPS, etc.) when connecting outside of company network. (3)
- Receivers do not need to be online or waiting for the messages. Tool can save messages for later delivery. (3)
- The tool keeps all the authentication credentials protected from unauthorized access. (5)
- Different access levels to various controls are possible. (6)
- The tool can have a failover copy of itself on another server that takes over when main application goes offline. (6)

- Automated message reprocessing is possible. (7)

4.5.4. Connectivity

G&F is using many different applications to serve many different purposes and need to also communicate with other companies and their applications. There are many different ways all of these applications allow connections to be made and not all provide the same possibilities of connectivity. Some might require files to be saved to a folder inside company or over FTP to an external server, some might use message queues and some might require web services. No matter what the method of connection is, the EAI tool should be capable of connecting to applications and communicate with them.

As there can always emerge new ways of connecting applications together or there might be some ways that were not taken into account, customizability is very important part of connectivity. Customizability should enable new methods of connection to be created when needed.

- The tool is able to share data by using files. (1)
- The tool is capable of connecting over network using HTTP and FTP protocols (1)
- The tool is capable of using web services. (3)
- The tool is able to connect to database for purposes of data retrieval/insertion. (4)
- The tool is capable of connecting over network using some message queue standard. (4)

4.5.5. Performance/Scalability

Performance of the tool was of course important for G&F. They had thousands of messages moving every day and some messages were extremely time critical. Luckily most time critical messages were relatively small in size so only the amount was a problem with the time critical messages. There are some large messages that should also be processed by the EAI tool. These messages however, were not in any way time critical.

In cases where customer would need extra boost, the EAI tool should be capable to scale up to use multiple servers. Another option would be to install multiple instances of

the tool to separate servers. G&F was prepared to purchase extra servers to make the message processing go faster.

- The tool is capable of processing message sizes up to 20MB. (1)
- The tool is capable of processing multiple messages at the same time. (1)
- The tool is capable of processing about 40 000 messages with none or light message transformations per day (~1msg/2s) without problems. (2)
- The tool can be scaled to multiple servers. (5)
- Multiple instances of the tool can work at the same time on different servers inside the company network. (6)

4.5.6. Management/Monitoring

As with any enterprise application the EAI tool would need to provide possibility to manage and monitor its actions i.e. message traffic, application status, logs, errors and so on. Although most modern commercial applications do provide the graphical user interface, it was not seen as a mandatory requirement for the monitoring tool as long as using it would be clear and simple. If the monitoring could be done from the Jeeves ERP by embedding the EAI tool's monitoring view to it would be a great extra but not a critical feature.

Of course, the more information that could be retrieved from the monitoring tool the better it would be. But also it was important to identify what information would be useful and what would not.

- Message traffic can be monitored in real time. (2)
- Messages in error state can be accessed in real time. (2)
- System and message logs can be monitored in real time. (2)
- Application status is available in real time. (2)
- The usage of different system resources (CPU, RAM, Number of threads etc.) is available in the monitoring tool. (3)
- Graphical presentations of different message/system states, message traffic and other system features are presented. (8)
- The monitoring of the tool can be embedded to another tool. For example the Jeeves ERP. (10)

4.5.7. Message Transformation

Since the tool needs to be capable of receiving and sending messages in a plethora of different formats it will also need to be able to change the format of the message before

forwarding it to the receiver. Sometimes the tool might need to append two data elements together, split one element into two or even add information to the message from an external source before forwarding the message. It is critically important that the tool has very extensive and easy to use message transformation capabilities to provide for a fast and reliable way of transforming the message inside the EAI tool.

- The tool is capable of transforming messages in one format to another (for example XML to CSV or EDI file format). (1)
- The tool is capable of editing the message data before delivery to receiver. (1)
- The tool can add information to the message before delivery. (3)
- The tool is capable of splitting one message into multiple smaller messages. (4)
- The tool is capable of retrieving data from external sources and adding it to a message before delivery. (4)
- The tool is capable of combining multiple messages into one. (5)
- The message transformation component is fairly easy to use. (5)

4.5.8. Routing

According to G&F, in some situations the EAI tool would be needed to choose where to send the message according to predefined properties. For example an invoice might need to be sent to a customer's different offices according to the country code. The sending party might be the same for all invoices and it might not be possible for them to send messages differently according to country code so it becomes the responsibility of the EAI tool to decide which message belongs to which receiving address.

Routing using predefined properties with wildcard-symbols was not seen as very important option though it was seen as very useful if the tool would provide such a feature. Most important of routing would be that properties could be freely chosen from any data or metadata that would exist in the message instead of choosing from predefined options.

No matter in what manner does the EAI tool process messages there's always messages that are more important than others and they need to be processed immediately. When these messages are competing of the same resources inside the application it should be possible to tell the application which messages are more important and which are less important so that the application knows which messages should be picked first from the queue or which messages should be allocated more resources.

- Routing options can be defined freely from available data. (2)
- A message can be delivered to multiple receivers. (2)

- The tool can choose different delivery locations according to message envelope data. (3)
- The tool can prioritize some messages to be more important than others and process them accordingly. (3)
- The tool can choose different delivery locations according to message data. (4)
- The tool is capable of rerouting messages based on data from external sources. (5)
- Routing can be done using wildcard-symbols. (9)

4.5.9. Pricing

As the potential customers buying Jeeves ERP are small and medium sized enterprises it might be hard to spend extra money for an integration tool in addition to already purchasing the Jeeves ERP which is going to be an expensive investment in itself. So the pricing of the tool must be thought out carefully. These requirements only concern the commercial products.

- The overall costs of the tool should be low. (1)
- The message transformation components do not cost much. (3)
- The pricing should be clear and simple. (3)

4.6. Tool Comparison

In this subsection each tool is compared to the requirements and judged whether it is a suitable tool for integrating with Jeeves.

4.6.1. IBM WebSphere MQ

The tool passed 91% (51/56) of the requirements without any problems. 5% (3/56) of the requirements were seen as passed although some issues were found with them. 4% (2/56) of the requirements were seen as failed.

As the tool is completely commercial there is no source code available for this tool. This is not a big problem in changeability. The monitoring of the tool was seen as quite extensive although a little bit complex. Luckily the good documentation will help with this. Also it seems quite difficult to embed the management tool to the ERP. This is only a small issue and is acceptable. The tool has a message transformation\mapping component but it seems quite difficult to use.

IBM WebSphere Message Broker is a viable option for an EAI tool. The complexity of installation and configuring could be made easier and it might be a little heavy for the

purpose of this case study. The licensing system of 12 months license seems both good and bad. Good as initial cost is lower but as the years go by the licensing costs will grow every year.

4.6.2. Mule ESB (Enterprise and Community)

The tool passed 91% (51/56) of the requirements without any problems and the rest 9% (5/56) were also seen as passed although some small issues were seen with them.

The tool passed all requirements for customizability without problems.

Some learning curve is required for creating custom extensions and features for the tool but this was seen as acceptable. Security against harmful attacks passed without problems. Message transaction security also passed but there were some requirements that required to be specifically configured for the tool so that the requirement would be met.

These requirements were:

- The tool saves messages in case of system crashes so that it can return processing after the tool becomes online again.
- Receivers do not need to be online or waiting for the messages. Tool can save messages for later delivery.
- Automated message reprocessing is possible.

In the first one it's required that the queues used are set to persistent so that all messages are saved to memory instead of cache. Second requirement depends on which delivery protocol is used. For the automated reprocessing it needs to be known on which errors the message should be automatically reprocessed and what actions should be taken before it can be reprocessed. Reprocessing all messages every time would just clog up the processing queues as an erroneous message might get reprocessed eternally.

Problem found with transformation was with the requirement for EDI messages. The tool does not support EDI format. This is relatively big problem as it is known that at least some of the customers will require EDI format for communication. Fortunately this should be fixed within the current year of 2013 as Mulesoft has received similar complaints from multiple of its customers. Meantime Merit FI has a tool that can be used for EDI transforms when needed. The pricing of the tool was seen potentially as a little high for some smaller customers but not so high that it would remove the tool from being a potential candidate.

All in all Mule ESB seems like a very fitting tool for the integration needs that Merit FI has. All of the requirements are fulfilled to some extent and the tool itself has aroused some interest in some of the people working inside the company.

Although some learning curve is required for the tools it provides, it is to be expected with any new applications so this shouldn't be too much of a problem. Since it seems clear that MuleSoft tries to push the commercial version to their customer and the community version is clearly lacking in many important features it would require the customers to commit to the commercial route with this tool.

4.6.3. WSO2

The tool passed 96% (54/56) of requirements without problems. The 4% (2/56) percent were also seen as passed although small issues were found with a couple of requirements.

No issues were found in changeability but instead it seems that the documentation for the tool is very good. It could be argued that the tool's documentation is on the level that is expected from commercial products. Community around the tool also seems to be quite active. With performance it is worth mentioning that in the documentation about performance tuning it says that the production deployment for the tool is recommended for servers running Unix/Linux. This is a slight problem since the servers used will most likely be running windows operating system. Also, it seems there are some difficulties running the tool as a service on windows platforms. It seems there are relatively good options for creating message transformations. Mostly these seem to require some amount of coding to be done. Some learning curve is expected to creating transformations.

As an open source tool the WSO2 ESB is seen to be very close to commercial tools with their graphical user interfaces and configuration tools. Only difference seems to be the missing data transformation/mapping tool. Still, the tool requires some amount of configuration and quite a lot of learning on how to use the web console. Luckily to help in this there is an active community and excellent documentation to help in the learning process.

4.6.4. ServiceMix

The tool passed 77% (43/56) of the requirements without problems. 16% (9/56) of the requirements were seen as passed although some issues were found with them. Mostly these issues were configuration related. The tool failed 7% (4/56) of the requirements.

This was due to missing functionality or the problem was that enabling functionality required more work than was seen reasonable.

Some small stability issues were found with the tool's error handling capabilities. Mostly the problem was that the amount of configuration and knowledge about what should be configured was seen as quite a lot of work. With changeability a small issue found was that in order to add new functionality to the tool there was quite a large learning curve. Although the Open Source community around the tool seems somewhat active so any help required could probably be provided by the community. The management and monitoring of the tool was seen as one of the biggest problems in the tool. Since this was a major part in the requirements it would reduce the tool's points quite a lot. Much of the monitoring and management functionalities were found to be present in the tool but to use them would require knowledge of different technologies (JMX, Apache Karaf) which might demotivate people with less of a technical background. The web management tool that is provided as a feature does little to simplify the monitoring process as it mostly provides a static list of existing features and bundles that could then be installed or uninstalled from the tool. Most of the required functionalities could probably be included into a single tool but this would require more work than is seen reasonable.

The tool has no specially designed data transformation tool. Instead it relies on some common data processing protocols and languages (XSLT, JAXB, JSR223). This kind of approach requires much more knowledge and/or studying from the administrators. Also as this basically means that the administrators need to be capable of programming using one or all of these different ways it might not look like a tempting solution for them. Luckily Merit FI has been using a tool for creating data transformations and mappings this problem can probably be circumvented by using this external transformation tool.

The tool does fulfill most of the requirements with some success and issue, where some problems are found can mostly be compensated by using an existing solution. The most major problems with the tool are: Management/Monitoring which is seen as greatly lacking compared to some commercial tools that have a simple graphical management tool which includes most of the required management/monitoring needs. The amount of extra configuration needed to get everything working as is hoped for and the learning curve to get familiar with using the tool which is quite big and can be somewhat daunting for users with not a very technological background.

4.7. Recommendation for suitable tool

Any of these tools would be sufficient for the needs of G&F although some of these tools are more suitable than others. IBM WebSphere Message Broker is a somewhat monolithic tool for the required environment and is not the most recommended tool by this study. ServiceMix requires quite a lot of studying and configuring to get everything working as required. Also the monitoring of the tool is not very clear and using this tool requires quite a lot of technical background to be comfortable working with different user interfaces and configuration files. Although one of the most interesting tools studied, ServiceMix cannot be recommended as the integration tool for Merit FI and G&F.

Mule ESB attracted some interest from Merit FI during this study and it seems to be one of the most suitable tools from the list. Although only the commercial version can be recommended since the free version was seen lacking a lot of the required features and should not be used. WSO2 ESB was the tool that got the best score from passing the requirements and seems like the most suitable tool for the job. Although it did lack the mapping tool and like ServiceMix requires quite a lot of studying and configuration but with an active community and good documentation this should not prove to be a problem. The tools Mule ESB and WSO2 ESB will be recommended to Merit FI as the tools that should be used in their scenario as these seem to be the most suitable ones.

Apart from the EAI tools studied in this paper, there probably are many more tools that would've been equally suitable than the ones recommended but did not fit into the scope of this research.

5. Conclusions and Future Research and Development

Choosing the correct tool for connecting applications is not always easy. First of all it needs to be established whether or not a tool is even needed for connecting applications. When the number of integration points is, for example, under 5 and there is no expectation that the number will be increasing, spending all the time for choosing the right tool and creating integrations will probably just waste time from doing something more important. But when the number of integration points grows to more than 5 it should be considered whether an EAI tool should be acquired.

There is a large number of tools in the market and finding the right one can be challenging. A seemingly easy solution is to purchase a tool that is widely known and recommended. However, this kind of tools can be relatively expensive and it might not fit the special case of the company resulting in further purchases to “fill in the gaps”. It is best to spend some time researching for a right tool so that it can be made clear that the chosen tool fulfills the requirements of the company. Also it is good to include some free or open source tools to the list of possible options since nowadays many of free or open source applications can be equally good or even better than the commercial counterparts. An interesting point to notice is that two of the favorite tools of the author before writing this thesis ended up being some of the least suitable tools to recommend. As such it is very important to concentrate carefully on testing and reviewing the results so as to prevent the testers own biases to interfere in the selection process.

The case study used in this thesis represents a company trying to find a tool for an ERP solution that focuses on companies in the range of SME's. As the size of the company increases, other areas of requirements might arise and some presented here could become unnecessary. Also companies working in branches of industry not included in the scope of Jeeves ERP might find that there exist other more important areas that have not been included in this thesis. In a case study of the research conducted by Alshawi et al. [2004] the company found that IBM's MQSeries was the most suitable tool for integration. However the company in the case study was much larger and working in the telecommunications industry so the results are not completely comparable. Also they do not mention that any open source tools were included in the list of EAI-tools.

So, what to look in a tool? What are the qualities that should be paid attention to?

In this study we established that there are 9 different qualities that are important to an EAI tool. Those are Stability, Changeability, Security, Performance/Scalability, Connectivity, Management/Monitoring, Message Transformation, Routing and Pricing.

The importance of these qualities differs from case to case but a tool that has all these qualities should be a well-rounded and capable EAI tool for any situation.

As future work the research could be expanded to include the complete EAI tool selection process instead of just the requirements for selection.

References

- [Acsonnet, 2013] Acsonnet, ERP software, <http://www.acsonnet.com/erp.htm>, 2013.
- [ActiveMQ, 2013] ActiveMQ, Message Oriented Middleware, <http://activemq.apache.org/>, 2013.
- [Alshawhi et al., 2004] Sarmad Alshawhi, Marinos Themistocleous & Rashid Almadani, *Integrating diverse ERP systems: a case study*, The journal of enterprise information management, Vol. 17, Is. 6, 2004, 454–462.
- [Atom, 2005] Atom, XML-based Web content and metadata syndication format, <http://tools.ietf.org/html/rfc4287>, 2005.
- [Attarha & Modiri, 2011] Mina Attarha & Nasser Modiri, Focusing on the importance and the role of requirements engineering, in Proceedings of 4th International Conference on Interaction Sciences, 2011, 181-184.
- [Axis, 2010] Axis, Web Services/SOAP/WSDL engine, <http://axis.apache.org/>, 2010.
- [Axway, 2013] Axway B2Bi, Business to business integration solution, <http://www.axway.com/products-solutions/b2b/b2bi-solutions>, 2013.
- [Bouman et al., 1999] Jacques Bouman; Jos Trienekens & Mark van der Zwan, Specification of service level agreements, clarifying concepts on the basis of practical research, in proceedings of STEP '99: Software Technology and Engineering Practice, 1999, 169-178.
- [Brooks-Bilson, 2007] Rob Brooks-Bilson, The Evolution of integration architecture: An introduction to the enterprise service bus (ESB), <http://rob.brooks-bilson.com/index.cfm/Enterprise-Architecture>, 2007.
- [Burton, 2012] Katie Burton, Designed with SaaS providers in mind: Mule iON SaaS edition now available, <http://blogs.mulesoft.org/designed-with-saas-providers-in-mind-mule-ion-saas-edition-now-available/>, 2012.
- [Camel, 2013] Camel, Integration framework, <http://camel.apache.org/>, 2013.

- [CLR, 2013] CLR, Common Language Runtime, <http://msdn.microsoft.com/en-us/library/8bs2ecf4.aspx>, 2013.
- [Cope, 2010] Rod Cope, Comparison-of-Open-Source-ESB-Solutions, <http://www.openlogic.com/resources-library/?Tag=Webinars>, 2010.
- [Curry, 2004] Edward Curry, Message-Oriented Middleware, *Middleware for Communications*, John Wiley & Sons Ltd, 2004, 1-26.
- [Cymatric, 2011] Cymatric, Enterprise Application Integration, <http://cymatric.com/eai.html>, 2013.
- [CXF, 2013] CXF, An Open-Source Services Framework, <http://cxf.apache.org/>, 2013.
- [Eclipse, 2013] Eclipse, Integrated development environment, <http://www.eclipse.org/>, 2013.
- [EDI, 2013] EDI, Electronic data interchange, <http://www.unece.org/trade/untidd/welcome.html>, 2013.
- [FTP, 2013] FTP, File Transfer Protocol, <http://www.w3.org/Protocols/rfc959/>, 2013.
- [Fuse, 2013] Red Hat JBoss Fuse, Open source enterprise service bus, <http://www.redhat.com/products/jbossenterprisemiddleware/fuse/>, 2013.
- [Hohpe & Woolf, 2005] Gregor Hohpe & Bobby Woolf, Enterprise Integration Patterns, Addison Wesley, 2005.
- [Hughes & Cotterell, 2009] Bob Hughes and Mike Cotterell, Software Project Management, McGraw-Hill Education, 2009.
- [HTTP, 1999] HTTP, Hypertext Transfer Protocol, <http://www.w3.org/Protocols/rfc2616/rfc2616.html>, 1999.
- [IBM, 2011] IBM, WebShpere Message Broker, <http://www14.software.ibm.com/webapp/download/search.jsp?pn=WebSphere+Message+Broker>, 2011.

- [ISO/IEC 9126, 1991] Information technology – Software product evaluation – Quality characteristics and guidelines for their use, ISO/IEC 9126, 1991.
- [JBoss, 2013] JBoss ESB, Open source enterprise service bus, <http://www.jboss.org/jbossesb>, 2013.
- [Jeeves, 2013] Jeeves, Enterprise resource planning system, <http://www.jeeves.se/en/>, 2013.
- [JMS, 2002] JMS, Java Message Oriented Middleware - API, <http://www.oracle.com/technetwork/java/docs-136352.html>, 2002.
- [JMX, 2006] JMX, The Java Management extension, http://docs.oracle.com/javase/7/docs/technotes/guides/jmx/JMX_1_4_specification.pdf, 2006.
- [JSR208, 2005] JSR208: Java Business Integration, <http://jcp.org/en/jsr/detail?id=208>, 2005.
- [JSR222, 2009] JSR222, Java Architecture for XML Binding (JAXB), <http://jcp.org/en/jsr/detail?id=222>, 2009.
- [JSR223, 2006] JSR223, Scripting for the Java Platform, <http://www.jcp.org/en/jsr/detail?id=223>, 2006.
- [Karaf, 2013] Karaf, Lightweight OSGi based runtime container for component and application deployment, <http://karaf.apache.org/>, 2013.
- [LDAP, 2006] LDAP, Lightweight Directory Access Protocol, <http://tools.ietf.org/html/rfc4510>, 2006.
- [Manouvrier & Menard, 2010] Bernard Manouvrier & Laurent Menard, Application Integration: EAI, B2B, BPM and SOA, John Wiley & Sons, 2010.
- [Matjaz et al., 2010] B. Juric Matjaz, Chandrasekaran Swami & Frece Ales, WS-BPEL 2.0 for SOA Composite Applications with IBM WebSphere 7, Packt Publishing Ltd, 2010.

- [Malhotraa & Temponi, 2010] Rajiv Malhotraa & Cecilia Temponi, *Critical decisions for ERP integration: Small business issues*, International Journal of Information Management, Vol 30, 2010, 28-37.
- [Merit FI, 2013] Merit Consulting Finland, IT Consulting company, <http://www.meritglobe.com/fi/>, 2013.
- [MIME, 2013] MIME, Multipurpose internet mail extensions, <http://www.faqs.org/rfcs/rfc1521.html>, 2013.
- [MuleSoft, 2013a] Mulesoft, Enterprise service bus, <http://www.mulesoft.com/mule-esb-enterprise>, 2013.
- [MuleSoft, 2013b] Mulesoft, Enterprise service bus, <http://www.mulesoft.com/high-availability-mule-esb>, 2013.
- [MuleSoft, 2013c] MuleSoft, What is Mule ESB?, <http://www.mulesoft.org/what-mule-esb>, 2013.
- [MuleSoft, 2013d] Mulesoft, Documentation, <http://www.mulesoft.org/documentation/display/current/How+to+Use+the+Management+Console>, 2013.
- [MuleSoft, 2013d] MuleSoft, Understanding Enterprise Application Integration - The Benefits of ESB for EAI, <http://www.mulesoft.com/enterprise-application-integration-eai-and-esb>, 2013.
- [Oakland, 2013] Oakland, A data transformation tool, <http://oaklandsoftware.com/data-transformer>, 2013.
- [ODE, 2013] ODE, Orchestration Director Engine, <http://ode.apache.org/>, 2013.
- [OpenESB, 2013] OpenESB, The open enterprise service bus, <http://www.open-esb.net/>, 2013.
- [OSGi, 2013] OSGi, Set of specifications that defining a dynamic component system for Java, <http://www.osgi.org/Main/HomePage>, 2013.

- [Patrick, 2002] Jon David Patrick, Mohammed A. Rashid, Liaquat Hossain, *Enterprise Resource Planning: Global Opportunities & Challenges*, IGI Global, 2002.
- [QoS, 1997] QoS, Defines the requirements for network elements that support guaranteed service, <http://tools.ietf.org/html/rfc2212>, 1997.
- [Reca, 2011] Mateo Almenta Reca, New in Mule 3.1 Enterprise, <http://blogs.mulesoft.org/new-in-mule-3-1-enterprise/>, 2011.
- [REST, 2013] REST, Style of software architecture for distributed systems, https://en.wikipedia.org/wiki/Representational_state_transfer, 2013.
- [Rinneheimo, 2008] Rami Rinneheimo, Järjestelmäintegraatio ja asynkroninen tiedonsiirto, M.Sc. thesis, University of Tampere, 2008.
- [Salil & Swaminathan, 2010] Ahuja Salil & Chandrasekaran Swaminathan, Application development For IBM Websphere Process Server 7 and Enterprise Service Bus 7: Build Soa-Based Flexible, Economical, And Efficient Applications, 2010.
- [ServiceMix, 2013] ServiceMix, Open-source integration container, <http://servicemix.apache.org/docs/4.4.x/user/index.html>, 2013.
- [Smooks, 2012] Smooks, an extensible framework for building applications for processing XML and non XML data using Java, <http://www.smooks.org/>, 2012.
- [SMTP, 2008] SMTP, The basic protocol for Internet electronic mail transport, <https://tools.ietf.org/html/rfc5321>, 2008.
- [SOA, 2013] Service Oriented Architecture, SOA, <http://www.servicetechspecs.com/soa>, 2013.
- [SOAP, 2007] SOAP, A lightweight protocol intended for exchanging structured information in a decentralized, distributed environment, <http://www.w3.org/TR/soap12-part1/>, 2007.
- [SQL, 2013] SQL, A special-purpose programming language designed for managing data held in a relational database management system, http://www.iso.org/iso/catalogue_detail.htm?csnumber=45498, 2013.

- [StreamServer, 2013] StreamServer, Enterprise information management software, <http://www.streamserve.com/>, 2013.
- [SWIFT, 2013] SWIFT, Society for worldwide interbank financial telecommunication, <http://www.swift.com>, 2013.
- [Synapse, 2012] Synapse, Lightweight ESB, <http://synapse.apache.org/>, 2012.
- [Tacton, 2013] Tacton, Tacton Systems AB, <http://www.tacton.com/>, 2013.
- [TCP/IP, 2013] TCP/IP, A large collection of different communication protocols based upon the two original protocols TCP and IP, http://www.w3schools.com/tcpip/tcpip_protocols.asp, 2013.
- [Themistocleous et al., 2004] Marinos Themistocleousa, Zahir Irania & Peter E.D Love, *Evaluating the integration of supply chain information systems: A case study*, European Journal of Operational Research, Vol. 159, Is. 2, 2004, 393–405.
- [Umapathy & Burao, 2005] Karthikeyan Umapathy & Sandeep Burao, Designing Enterprise Solutions with Web Services and Integration Patterns, *IEEE International Conference on Services Computing*, 2006, 111 - 118.
- [URL, 2005] URL, A specific character string that constitutes a reference to a resource, <http://www.w3.org/Addressing/>, 2005.
- [WS-BPEL, 2007] WS-BPEL, Web services business process execution language, <http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.html>, 2007.
- [WSO2, 2013] WSO2 ESB, Enterprise service bus, <http://wso2.com/products/enterprise-service-bus/>, 2013.
- [XML, 2012] XML, A markup language defining rules for human-readable and machine-readable document format, <http://www.w3.org/XML/>, 2012.
- [XSLT, 1999] XSLT, A language for transforming XML documents into other XML documents, <http://www.w3.org/TR/xslt>, 1999.
- [XQuery, 2010] XQuery, An XML query language, <http://www.w3.org/TR/xquery/>, 2010.